Natural Language Processing

Lab 2: Preprocessing Natural Text

Objective

To understand text preprocessing and text normalization, including techniques like tokenization, stemming and lemmatization.

Rules

Non-compliance with the following rules will result in serious consequences.

1. Students who are more than 15 minutes late are not allowed to sit in the lab.
2. Monitors will remain off until prompted.
3. Complete silence in the lab.
4. Do not speak in between lectures.
5. For any queries, stay silent and raise your hand.
6. Phones shut off or on vibration mode during lab.
7. Labs will be submitted in hard copy.
8. The previous lab will be submitted in the first 15 minutes of the lab.
9. No late submissions will be accepted.
10. Treat lab tasks as your quiz or lab assignment.
11. No usage of A.I. for lab assignments or lab work.

Text Normalization

Text normalization is the process of transforming text into a single canonical form that it might not have had before. Normalizing text before storing or processing it allows for separation of concerns, since input is guaranteed to be consistent before operations are performed on it. These include:

- Removal of non-natural text, such as URLs and emojis
- Decoding HTML entities.
- Replacing substrings.
- Setting the casing of text.

Lab Task 1

Write a function `remove_multispace(sentence)` that takes a string as input and returns the string with multiple spaces replaced with a single space and removes leading and trailing whitespace.

- Input: “This string has a lot of spaces for some reason. ”
- Output: “This string has a lot of spaces for some reason.”
- Apply this function to a small paragraph (3-4 sentences) and display the original and processed text.

Lab Task 2

Write a function decode_html(sentence) that takes a string as input and returns the string with HTML entities replaced with canonical symbols. Do not use Python’s html library.

- Input: “This is a < sign.”
- Output: “This is a < sign.”
- Apply this function to a small paragraph (3-4 sentences) and display the original and processed text.

Stop Words

Stop words are generally words that are not considered to add information content to the question at hand. These are most commonly articles, prepositions and conjunctions.

Lab Task 3

Write a function `remove_stopwords(sentence)` that takes a string as input and returns the string with all stop words removed.

- Input: “The quick brown fox jumps over the lazy dog”
- Expected Output: “quick brown fox jumps lazy dog”
- Apply this function to a small paragraph (3-4 sentences) and display the original and processed text.

Natural Language Tool Kit (NLTK)

A suite of libraries and programs for symbolic and statistical natural language processing (NLP) for English written in Python. It supports processing tasks such as the following.

1. Classification
2. Tokenization
3. Stemming
4. Lemmatization
5. Tagging
6. Parsing
7. Semantic reasoning

Stop Words with NLTK

```
import nltk
nltk.download('stopwords')

from nltk.corpus import stopwords

stop_words = stopwords.words('english')
```

Lab Task 4

Do Lab Task 3 via NLTK. Note the time difference between the two tasks.

Tokenization

It is the process of converting a sequence of text into smaller parts, known as tokens.

Types of Tokenization

There are four types of tokenization.

1. Character tokenization
2. Word tokenization
3. Subword tokenization
4. Sentence tokenization

Lab Task 5

Write a function that tokenizes a sentence by splitting on whitespace.

Tokenization in NLTK

```
import nltk
nltk.download('punkt_tab')

from nltk.tokenize import word_tokenize
from nltk.tokenize import sent_tokenize

sent = “This is a sentence. This sentence is an example string.”
print(word_tokenize(sent))
print(sent_tokenize(sent))
```

Lab Task 6

Do Lab Task 5 via NLTK. Note the time difference between the two tasks.

Stemming

Stemming is a technique used to reduce an inflected word down to its word stem. For example, the words “programming,” “programmer,” and “programs” can all be reduced down to the common word stem “program”.

Lab Task 7

Implement a function `simple_stem(word)` that applies the following rules in order:

1. If the word ends with "ing", remove it.
2. If the word ends with "s", remove it.
3. If the word ends with "ed", remove it.

Stemming in NLTK

```
from nltk.stem import PorterStemmer

ps = PorterStemmer()
ps.stem('computer')
```

Lab Task 8

Do Lab Task 7 via NLTK. Note the time difference between the two tasks.

Lemmatization

Lemmatization is a technique used to reduce inflected words to their root word.

Lemmatization in NLTK

```
from ntlk.stem import WordNetLemmatizer

l = WordNetLemmatizer()
l.lemmatize(“computer”)
```

Lab Task 9

Write a program that takes a string as input, normalizes it, removes stop words, tokenizes the normalized string by words and lemmatizes all the words.